# Java Cryptography

* **JCA**: Java Cryptographic Architecture
* **JCE**: Java Crupthograhic Encryption

From JDK 1.4 onwards JCE is part of JDK.

The JCA within the JDK includes two software components:

This framework includes four packages

1. javax.crypto.\*
2. java.security.\*
3. javax.crypto.spec.\*
4. javax.crypto.interfaces.\*

Actual providers such as Sun, SunRsaSign, SunJCE, which contain the actual cryptographic implementations

Whenver a specific JCA provider is mentioned, it will be referred to explicitly by the provider’s name.

Architecture

**Cryptographic Service Providers (CSP)**

*Javax.security.Provider* is the base class for all security providers.

Each CSP contains an instance of this class that contains the provider’s name and lists all of the security services/algorithms it implements.

When an instance of a particular algorithm is needed, the JCA framework consults the provider’s database, and if a suitable match is found, the instance is created.

Public-key encryption is asymmetric. In asymmetric cryptography two keys are required: a public key that encrypts plain text into cipher text, and a private key that decrypts the cipher text back into plain text. This is in contrast to symmetric cryptography where there is a single private key that encrypts and decrypts.

In a single domain, like a website or application, symmetric encryption is often sufficient. The same party encrypting is typically the same party decrypting. However, asymmetric encryption is preferred when dealing with multiple parties or domains.

 Bruce Schneier’s *Applied Cryptography* (Wiley)

A mathematics ofcryptography is the Handbook of AppliedCryptography by Alfred J. Menezes et al. (CRC Press).

The javax.crypto package is the heart of the JCE. It defines the SecretKey interface.

Which extends java.security.Key to provide support for symmertric, secret-key cryptography.

The classes define by javax.crypto also define fundamental cryptographic objects.

They are summarized as follows:

**Cipher**: Encapsulates the notion of a cipher and provides support for cipher selection, encryption and decryption.

**CipherSpi**: The service provider interface for the Cipher class.

**NullCipher**: A subclass of Cipher that maps plaintext to ciphertext without peforming any transformation. It is referred to as an identity cipher and is typically used for testing purposes.

**CipherInputStream**- The combination of an InputStream and a Cipher that supports the decryption of data that is read from the InputStream using the Cipher

**CipherOutputStream**-The combination of an OutputStream and a Cipher that supports the encryption of data that is writtern to the OutputStream using the Cipher